ARMY RESEARCH LABORATORY

The Tank Accuracy Model

Fred L. Bunn

ARL-MR-48 March 1993

NOTICES

Destroy this report when it is no longer needed. DO NOT return it to the originator.

Additional copies of this report may be obtained from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Road, Springfield, VA 22161.

The findings of this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

The use of trade names or manufacturers' names in this report does not constitute indorsement of any commercial product.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1993	Final, 1 Jan 92 - 31 Oct 92

4. TITLE AND SUBTITLE
The Tank Accuracy Model

5. FUNDING NUMBERS
PR: 1L162618AH80

6. AUTHOR(S)
Fred L. Bunn

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: AMSRL-WT-WE
Aberdeen Proving Ground, MD 21005-5066

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: AMSRL-OP-CI-B (Tech Lib)
Aberdeen Proving Ground, MD 21005

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ARL-MR-48

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report discusses the theory of tank gun accuracy and the Tank Accuracy Model (TAM) software for generating accuracy data. It is designed for those who produce and use such data for current weapon systems as well as for those who wish to analyze future tank firepower systems. The model produces data for stationary armored vehicles firing at stationary targets and can provide the stationary/stationary data for input to stationary/moving models. The TAM software operates on any IBM PC compatible computer with a Fortran 77 compiler and generates accuracy data in a few minutes.

14. SUBJECT TERMS
Armored vehicles, tanks (combat vehicles), accuracy, cannons, hit probabilities

15. NUMBER OF PAGES
54

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

INTENTIONALLY LEFT BLANK.

ACKNOWLEDGEMENTS

The following people contributed their advice and time to improve this report:

Mr. Richard Norman, AMSAA, supplied test data and expert advice about methods of calculating error components. Mr. Edward Schmidt, ARL, reviewed the report and suggested numerous improvements. Mr. Joseph Olah, ARL, advised on the organization of the report and made helpful suggestions. Mr. Francis Mirabelle, ARDEC, supplied drag data which was used to test the model. Their assistance is much appreciated.

INTENTIONALLY LEFT BLANK.

CONTENTS

INTENTIONALLY LEFT BLANK.

LIST OF FIGURES

INTENTIONALLY LEFT BLANK.

LIST OF TABLES

INTENTIONALLY LEFT BLANK.

1. INTRODUCTION

The Tank Accuracy Model (TAM) generates accuracy data and hit probabilities for a stationary tank firing at a stationary target. It reads control values, ballistic and atmospheric data, and data concerning the fire control system. These are used to generate the fixed biases, variable biases, round to round dispersions and hit probabilities against a standard 2.3 by 2.3 meter standard NATO target.

This chapter serves as an introduction to the code and as a users' manual for the software. It gives introductory information and discusses the input and output of the program. The remaining chapters serve as a programmers' manual. They discuss the theory of tank gun accuracy and show how the theory is implemented in code.

The software currently handles fully functional modern tanks. Damaged tanks or older tanks may have additional sources of errors not yet handled by the software.

TAM is about 600 lines of Fortran 77, written in top down structured form. It runs on any computer that has a compiler for the full Fortran 77 language. It does not tax the limits of an IBM compatible PC and runs in a few minutes. It is a rewrite of the AMSAA PH1 program. A series of four test cases showed that both programs produce the same hit probabilities to three decimal places. For more details comparing the programs, see Appendix A.

What is the program good for? It is useful for producing typeset quality summary tables for publication and distribution. It is useful for producing computer readable input to other weapon system analysis programs. And it is useful for analyzing conceptual systems with smaller error budget components.

Summary Output. Table 1 illustrates the summary output. The title line contains blanks because the table would often be classified if the blanks were filled in. The first column contains the range from the firer to the target. The next eight columns contain the errors in mils. The final column contains the probability of hitting the 2.3 by 2.3 meter standard NATO target. The Total dispersion columns are the root sum squares of the preceeding two columns.

TABLE 1. Sample of Summary Output After Typesetting

Table ___. Ballistic Errors for a _____ Round Fired from a Stationary _____ with a _____ Range finder and a Computer

			Horizontal (mils)			Vertical (mils)			
Range (m)	Fixed Bias (mils) horiz.	vert.	Ran. error	Var. bias	Total disp.	Ran. error	Var. bias	Total disp.	P_h
500.	0.0000	-0.2100	0.6937	0.3955	0.7985	0.6907	0.4150	0.8058	0.992
1000.	0.0300	-0.1800	0.4130	0.4268	0.5939	0.4130	0.4066	0.5795	0.900
1500.	0.1100	-0.1200	0.3294	0.4969	0.5962	0.3358	0.4258	0.5423	0.674
2000.	0.0500	-0.0600	0.2919	0.6495	0.7121	0.3219	0.4756	0.5743	0.405
2500.	0.0000	0.0000	0.2716	0.8403	0.8831	0.3287	0.5631	0.6520	0.213
3000.	-0.0500	0.0600	0.2592	1.0747	1.1055	0.3452	0.7197	0.7982	0.103

Installation. This office normally supplies programs on IBM compatible floppy disks. The program is written in a mix of upper and lower case for readability and to highlight branches and loops. It also contains a tab on each line to space over to 'column 7'. The Unix tr command may be used to switch to all upper case and the Unix expand command to replace tabs with spaces. If desired, the code can be supplied in all upper case without tabs. The only other nonstandard feature the author is aware of is a call to the fdate function in the main program. If your Fortran compiler does not recognize this function, either use a work around or comment out the printing of the date. Compile the program so that it uses double precision. If there are any other problems, call us at (410) 278-6676.

1.1 Input Data

This section discusses how to prepare the input data. As appropriate it will point out standard values to use or sources for the data. Those not familiar with the various sources of error should consult later sections which document the code. Those sections explain why each error source causes a delivery error. The equations and code presented in the discussion may be helpful to users, but need not be understood to properly prepare the input.

Conventions. The firing tank is at the origin of the coordinate system and the target is at r_t,0,0. The x axis is positive east, the y axis is positive north, and the z axis is positive upward.

TAM uses metric units and radians internally. It reads temperatures in degrees Fahrenheit or centigrade and converts to degrees Kelvin. It reads angles as degrees or mils but uses radians internally. The angular errors produced are in mils. The mil is 1/6400 of a circle, the standard military unit of measuring angles for weapon systems.

The components of variable bias and random error are assumed to be independent, normally distributed random variates. They are aggregated by root sum squaring.

TAM reads all data in free format, so the numbers need not be in any particular columns (fields) of the input file. Values may be separated by spaces, commas, or tabs. Where a character string appears, it is to be treated as a single value. If it contains spaces, commas, or tabs, it may need to be enclosed in single quotes. Some TAM input consists of tables. These tables may have no more than 20 entries for each variable.

The program reads program control data, atmospheric data, ballistic trajectory data, and error budget data. The first four lines contain data for controlling the run. The fifth line contains atmospheric data. The next set of lines contains ballistic trajectory data. The number of lines is variable. The remaining set contains error budget data. The number of lines in this set is also variable.

Control information. The first line contains information about the data or run. Use it to document the contents of the input file and describe why you are performing this run. The program will print it with the output if you set it to echo input. This allows you to go back later and determine the purpose of the input and output.

The second line controls the type and amount of output. It contains four logical values. If they are all 'F', TAM prints no output. If the first is 'T', TAM prints an input

TABLE 2. Sample Input File

```
Test case 4. bc=1.34, vm=1140 m/s.
T T T F                    Print flags
4                          Ranges (m)
1000. 2000. 3000. 4000.
15., 1.2249992             Air temperature (C), density (kg/cu m).
1140.                      Muzzle velocity (m/s)
1.34                       Ballistic coefficient (lb/sq in)
10                         Drag data
   1.000   0.810
   1.050   0.850
   1.100   0.855
   1.150   0.850
   1.200   0.830
   1.500   0.750
   2.000   0.655
   2.500   0.570
   3.000   0.510
   4.000   0.400
10 FIXED BIAS
250 500. 800. 1000. 1200. 1500. 1600. 2000. 2500. 3000. 3500. 4000.
0     0    0    .03   .06   .11    .1   .05    .0  -.05  -.05 -.05
-.21 -.21 -.21 -.18  -.16 -.12  -.11  -.06   .00   .06   .06   .06
-2 CANT VALUES
5.0 0.5                    Cant (deg)
1.79832                    Crosswind (m/s)
10, JUMP VALUES
250 500. 800. 1000. 1200. 1500. 1600. 2000. 2500. 3000. 3500. 4000.
 .29 .29 .29  .28  .26  .24  .25  .27  .29  .29  .29  .29  .29  .29
 .37 .37 .37  .37  .38  .38  .38  .40  .42  .42  .42  .42  .42  .42
10, FIRE CONTROL VALUES
250. 500. 800. 1000. 1200. 1500. 1600. 2000. 2500. 3000. 3500. 4000.
 .15 .112 .075 .057 .046 .053 .055 .083 .127 .179 .18 .18 .18
 .156 .106 .075 .035 .032 .043 .055 .101 .167 .267 .27 .27 .27 .27
 .22 .15      Boresight retention (mils)
T 5. -.7271   Finned, range error (m), horiz gun/sight offset (m)
6.30936       Muzzle velocity variation (m/s)
5 m -.5276    Rg error (m or %), vertical gun/sight offset (m)
3.3528        Range wind (m/s)
4.44444       Air temp sigma (C)
 .015         Air density (fraction reduced)
0.0           Windage jump (mils)
0.03          Optical path bending (mils)
 .3 .05 0. 0.Lay error values
10, DISPERSION
250 500. 800. 1000. 1200. 1500. 1600. 2000. 2500. 3000. 3500. 4000.
 .21 .21 .21 .21 .21 .21 .21 .21 .21 .21 .21 .21
 .20 .20 .20 .21 .21 .22 .23 .25 .28 .31 .31 .31
```

echo. If the second is 'T', TAM prints intermediate input in an error budget table. If the third is 'T', TAM prints summary errors separated by spaces in a form suitable for input to other Fortran programs. If the last value is 'T', TAM prints summary errors

separated by tabs in a form suitable for input to the TROFF typesetting program (and its preprocessors EQN and TBL). This last form produces typeset quality output.

The third and fourth lines tell the program the target ranges at which you wish it to generate data. The third line contains the number of ranges and the fourth lists the ranges.

Atmospheric data. TAM needs to know the air temperature and density. Normally, you should use the standard value for air temperature. It is 15 degrees Centigrade (59 degrees Fahrenheit). The standard value for air density is 1.2249992 kilograms per cubic meter. These are average year round values at sea level. You may wish to change these values to evaluate a system at higher altitudes, or at nonstandard climates or seasons.

Round data. The next segment of input describes the ballistics of the round. This information may be obtained from the Firing Tables Branch of the Army Research Development and Engineering Center (ARDEC). It contains the muzzle velocity, ballistic coefficient, and drag data. The muzzle velocity is in meters per second. The ballistic coefficient is in pounds per square inch. It is the mass of the projectile divided by its cross sectional area. That is the area presented to the 'wind'. The final data is a table defining the C_d drag as a function of mach number.

Format of remaining data. TAM next reads detailed error budget information. This section discusses the format of the data, describes the error sources, and tells where you can get data to feed the program.

If error data can be read in several formats, the first line of the data will contain an integer and a character string. Here's what the integer means:

Integer	Comment
<0	Compute the error source from special values that follow.
0	Leave these errors set to zero and go to next error type.
1	Set the error to constants at all ranges. The horizontal and vertical constants are the two values on the next line (mils).
$n>1$	Interpolate in a table having three rows and n columns. The first row is range, the second the horizontal or y values, and the third contains vertical or z values (mils).

If the data is read in only one way and that's not as a table, then the input data will be on a single line.

Fixed bias data. Normally, the fixed bias will be zero. In this case, replace the 10 in the example with a 0 and delete the next three lines. The next most common case is to read the fixed bias data in as a table, as illustrated in the example. It is possible to set the fixed biases constant for all ranges but it doesn't seem realistic. If there's a reason to do so, replace the 10 in the example with a 1 and replace the next three lines with a single line containing the horizontal and vertical fixed biases (mils).

The final option is to have the program compute the fixed bias by calculating the parallax error or drift error or both. When using this option, replace the 10 in the

example with -1. On the next line, put three values in meters. The first value is the expected range to the target. It should be 0.0 if the fire control accurately corrects for parallax. It should be the zeroing range (often 1200.0 m) if the fire control makes no correction for parallax. And it should be the battle sight range if battle sight ranging is in use. The second value is the horizontal offset of the gun from the sight and the third is the vertical offset.

Next, insert a line containing a 0, 1, 2, or 3. The 0 indicates that the round has no drift or the fire control corrects for drift. The other integers mean that the subsequent 5 coefficients give drift as a function of 1) range, 2) time of flight, or 3) super elevation. If the value chosen is 1, 2, or 3, insert five drift coefficients on the next line. These may be obtained from Firing Tables Branch of ARDEC. Use the parallax data and drift coefficients if the tank is a modern system with the appropriate components damaged or if the tank is an older tank that does not have parallax and drift correction.

Cant data. Normally, the cant error is calculated. When the firing platform is canted and the cant is unmeasured or imperfectly measured, it causes a delivery error that persists but has zero mean. The horizontal error and vertical errors are a function of the cant angle and super elevation angle. If cant is not measured, replace the -2 shown in the sample data with a -1 and on the next line place the standard deviation of cant. Five degrees is the standard value. If cant is measured, the -2 shown in the sample data is correct. The next line contains the standard deviation of cant (probably five degrees) and the standard deviation of measurement error.

Crosswind data. Normally, the crosswind error is calculated. The standard value for crosswind is 1.8 m/s.

Jump data. Jump is the residual systematic error remaining after all other systematic errors are accounted for. Jump is a function of target range and TAM reads it in as a table as shown in the example.

Fire control data. The fire control system is unable to point the gun exactly as computed. This is due in part to 'slop' in the mechanical linkages. The error varies with range and is read in as a table, as shown in the example.

Boresight data. Boresight error is a constant angle horizontally and vertically. Specify the errors by two angular values as shown.

Parallax and drift compensation data. When the fire control computer compensates for parallax and drift, it uses an imperfect measurement of target range. TAM reads the standard deviation of that measurement and finds the associated horizontal and vertical errors. The value for standard deviation may be given as a linear error (five meters is standard for a laser ranger), or it may be given as a fraction of the true range. If you are evaluating a system with a laser ranger, and want to use the normal value, five meters, for the standard deviation, just input '5.0 m'. The program will read the 'm' and treat the '5.0' as a linear error in meters. If you are evaluating a system with a coincident range finder and want to use a value of 20% of true range, just input '20. %'. The program will read the '%' and treat the error as 20% of the true range.

Muzzle velocity variation data. The propellant temperature of the rounds varies over time and is the major contributor to variations in muzzle velocity from

occasion to occasion. When the muzzle velocity is different from what is expected, the super elevation of the barrel will be wrong. Input the standard deviation of the muzzle velocity in meters per second.

Range error data. This is the standard deviation of error in measuring range. For laser range finders it's normally 5 meters. For other range finders it's a percentage. After the value, leave a space or tab then an 'm' or '%' to specify how the number is to be treated.

Air temperature data. This is the standard deviation of the air temperature in degrees centigrade. The standard value is 4.444 degrees centigrade (8 degrees Fahrenheit).

Air pressure data. This is the standard deviation of the barometric pressure. It is input as a fraction of the standard barometric pressure.

Windage jump data. When a spinning round leaves the barrel and cross wind is present, the cross wind imparts a force on the spinning round causing it to twist in the vertical plane. This results in a vertical error. For spinning rounds, use a non-zero value (mils), and for smooth bore cannon, use a zero error.

Optical path bending data. This is the standard deviation of the error and is normally 0.03 mils. Optical path bending is caused by the sunlight heating the air or the lack of sunlight cooling the air through which the light passes and refracts. This bending of the light causes the target to appear to be at a location different from where it actually is, to shimmer, or to appear broken. The significance of this error source has been debated for years. There are those who feel the error is not significant at two kilometers, significant at three kilometers, and very significant beyond three and a half kilometers. Others feel the error is significant at shorter ranges. During the day the optical path bends upward causing the gunner to hit low, and at night the optical path bends downward causing the gunner to hit high.

Lay error data. This line should normally contain the quasi-combat values: '0.3 0.05 0. 0.'. Models for stationary firer vs. moving target generate their own lay errors. If you wish to generate stationary / stationary data with no lay error for input to such a model, input four zero values on this line. To use other values, see section 4.4 for details.

Dispersion data. The dispersion data may be in two forms. Table 2 shows the first form. In this form, there are eight values for horizontal and vertical dispersion at each of eight ranges. This is the normal way to input this data. It is based on experimental data and includes the effect of variations in muzzle velocity from round to round. Such variations cause a vertical dispersion.

The alternate form assumes the dispersion is constant and calculates the effect of muzzle velocity variations from round to round. If you use this form, the data should look like this:

```
0          Dispersion data
 .21 .21   Horiz, vert (mils)
5.3        Std dev of muzzle velocity (rd-rd)
```

1.2 Input Echo

The first output TAM prints is an input echo (assuming the print flag is turned on). This lets you verify the input is in the proper format and provides an audit trail. The input echo also contains the date of the run and includes the super elevation and time of flight to the desired target ranges.

Table 4 illustrates the input echo. It should be very similar to the input file. The first major difference is that line 2 contains the date and time of the run. The second is lines 13-16 which contain the desired target ranges plus the super elevation in mils and the time of flight in seconds. TAM calculates these values rather than reading them.

1.3 Error Budget Output

If the second print flag is set, TAM prints intermediate tables. These contain the primary data TAM generates. They are useful for understanding the program, verifying that it is working properly, and may even be directly useful for your analysis of tank fire control systems.

Table 5 shows an example. Except for range, which is in meters, all values in the table are errors in mils. Note that lines two and three are all zeros in the example. This is the case when fixed biases are read as a table. Next, note that lines 12-17 in the Horizontal errors table are not printed. There are no values calculated for these lines. For the same reason, lines 16 and 17 in the Vertical errors table are not printed.

1.4 Summary Output

The summary output may be printed in one or both of two formats. The first (plain) format prints the data in a form suitable for input to another program. The second (fancy) format prints the data in a form suitable for typesetting.

The arrangement of the data is the the same for both the plain and fancy formats. Column 1 gives the range to the target in meters. Columns 2-9 give the errors in mils. The last column gives the hit probability on a standard NATO 2.3 by 2.3 meter target.

Table 3 shows the data in the plain format.

TABLE 3. Sample Summary Output for Input to Another Program

500.	0.0000	−0.2100	0.6937	0.3955	0.7985	0.6907	0.4150	0.8058	0.992
1000.	0.0300	−0.1800	0.4130	0.4268	0.5939	0.4130	0.4066	0.5795	0.900
1500.	0.1100	−0.1200	0.3294	0.4969	0.5962	0.3358	0.4258	0.5423	0.674
2000.	0.0500	−0.0600	0.2919	0.6495	0.7121	0.3219	0.4756	0.5743	0.405
2500.	0.0000	0.0000	0.2716	0.8403	0.8831	0.3287	0.5631	0.6520	0.213
3000.	−0.0500	0.0600	0.2592	1.0747	1.1055	0.3452	0.7197	0.7982	0.103

The fancy summary output is in a form suitable for processing by the Unix Troff and its Tbl pre-processor. Table 6 shows the output TAM generates for input to Troff. Typesetting instructions surround the title and data.

TABLE 4. Sample Input Echo

```
Test case 4. bc=1.34, vm=1140 m/s.
Sun Sep 20 16:00:24 1992
 T T T T   Print flags
TARGET RANGES (m)
    500.   1000.   1500.   2000.   2500.   3000.
  15.000   Air temperature (C)
   1.225   Air density (kg/cu m)
1140.000   Muzzle velocity (m/s)
   1.340   Ballistic coefficient (lbs/sq in)
DRAG DATA: Mach nr, KD
   1.000   1.050   1.100   1.150   1.200   1.500   2.000   2.500   3.000   4.000
   0.810   0.850   0.855   0.850   0.830   0.750   0.655   0.570   0.510   0.400
TRAJECTORY: range (m), super elev (mils), time of flight (sec)
    500.   1000.   1500.   2000.   2500.   3000.
   2.091   4.584   7.607  11.343  16.069  22.201
   0.467   1.000   1.615   2.336   3.194   4.230
FIXED BIAS DATA
    250.    500.    800.   1000.   1200.   1500.   1600.   2000.   2500.   3000.
   0.000   0.000   0.000   0.030   0.060   0.110   0.100   0.050   0.000  -0.050
  -0.210  -0.210  -0.210  -0.180  -0.160  -0.120  -0.110  -0.060   0.000   0.060
   5.000   Std. dev. of cant (deg)
   0.500   Std. dev. of cant sensor (deg)
   1.798   Cross wind (m/s)
JUMP VALUES
    250.    500.    800.   1000.   1200.   1500.   1600.   2000.   2500.   3000.
   0.290   0.290   0.290   0.280   0.260   0.240   0.250   0.270   0.290   0.290
   0.370   0.370   0.370   0.370   0.380   0.380   0.380   0.400   0.420   0.420
FIRE CONTROL VALUES
    250.    500.    800.   1000.   1200.   1500.   1600.   2000.   2500.   3000.
   0.150   0.112   0.075   0.057   0.046   0.053   0.055   0.083   0.127   0.179
   0.156   0.106   0.075   0.035   0.032   0.043   0.055   0.101   0.167   0.267
   0.220   Horizontal boresight value (mils)
   0.150   Vertical boresight value (mils)
   6.309   Muzzle velocity variation (m/s)
   5.000   Range Estimation (m)
  -0.528   Vertical gun / sight offset (m)
   3.353   Range wind value (m/s)
   4.444   Air temperature value (deg C)
   0.015   Air density (fraction reduced)
   0.000   Windage jump (mils)
   0.030   Optical path bending (mils)
RANDOM ERROR VALUES
   0.300   0.050   0.000   0.000Lay error constants
DISPERSION VALUES
    250.    500.    800.   1000.   1200.   1500.   1600.   2000.   2500.   3000.
   0.210   0.210   0.210   0.210   0.210   0.210   0.210   0.210   0.210   0.210
   0.200   0.200   0.200   0.210   0.210   0.220   0.230   0.250   0.280   0.310
```

TABLE 5. Sample Error Budget

HORIZONTAL ERRORS						
	RANGE (m)	500.	1000.	1500.	2000.	2500.	3000.
1	FIXED BIAS	0.0000	0.0300	0.1100	0.0500	0.0000	-0.0500
2	Parallax	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3	Drift	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
4	VARIABLE BIAS	0.3955	0.4268	0.4969	0.6495	0.8403	1.0747
5	Cant	0.0182	0.0400	0.0664	0.0990	0.1402	0.1937
6	Crosswind	0.1041	0.2247	0.3657	0.5328	0.7333	0.9762
7	Jump	0.2900	0.2800	0.2400	0.2700	0.2900	0.2900
8	Fire Control	0.1120	0.0570	0.0530	0.0830	0.1270	0.1790
9	Boresight	0.2200	0.2200	0.2200	0.2200	0.2200	0.2200
10	Plx, drf comp (PDC)	0.0148	0.0037	0.0016	0.0009	0.0006	0.0004
11	Rotation of Earth	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
18	RANDOM ERROR	0.6937	0.4130	0.3294	0.2919	0.2716	0.2592
19	Dispersion	0.2100	0.2100	0.2100	0.2100	0.2100	0.2100
20	Lay error	0.6612	0.3556	0.2537	0.2028	0.1722	0.1519

VERTICAL ERRORS							
	RANGE (m)	500.	1000.	1500.	2000.	2500.	3000.
1	FIXED BIAS	-0.2100	-0.1800	-0.1200	-0.0600	0.0000	0.0600
2	Parallax	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3	Drift	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
4	VARIABLE BIAS	0.4150	0.4066	0.4258	0.4756	0.5631	0.7197
5	Muzzle Velocity err	0.0237	0.0537	0.0926	0.1441	0.2145	0.3129
6	Rg est & PDC	0.0120	0.0247	0.0322	0.0411	0.0531	0.0700
7	Jump	0.3700	0.3700	0.3800	0.4000	0.4200	0.4200
8	Fire Control	0.1060	0.0350	0.0430	0.1010	0.1670	0.2670
9	Boresight	0.1500	0.1500	0.1500	0.1500	0.1500	0.1500
10	Range wind	0.0005	0.0029	0.0089	0.0214	0.0464	0.0958
11	Air temp	0.0019	0.0093	0.0259	0.0577	0.1154	0.2178
12	Air density	0.0028	0.0132	0.0362	0.0799	0.1584	0.2956
13	Windage jump	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
14	Optical path bend	0.0300	0.0300	0.0300	0.0300	0.0300	0.0300
15	Vertical cant	0.0017	0.0037	0.0061	0.0091	0.0128	0.0177
18	RANDOM ERROR	0.6907	0.4130	0.3358	0.3219	0.3287	0.3452
19	Dispersion	0.2000	0.2100	0.2200	0.2500	0.2800	0.3100
20	Lay error	0.6612	0.3556	0.2537	0.2028	0.1722	0.1519

TABLE 6. Sample Summary Output Before Typesetting

```
.ce 3
Table __. Ballistic Errors for a _____ Round Fired from a
Stationary _____ with a _____ Range finder and a
Computer
.TS
center box;
css|css|css|c
c|cs|ccc|ccc|c
c|cc|ccc|ccc|c
n|nn|nnn|nnn|n.
       Horizontal (mils) Vertical (mils)
_
Range  Fixed Bias (mils) Ran.   Var.   Total   Ran.   Var    Total   $P sub h#
(m)    horiz.   vert.    error  bias   disp.   error  bias   disp.
_
  500.  0.0000  -0.2100   0.6937 0.3955 0.7985  0.6907 0.4150 0.8058  0.992
 1000.  0.0300  -0.1800   0.4130 0.4268 0.5939  0.4130 0.4066 0.5795  0.900
 1500.  0.1100  -0.1200   0.3294 0.4969 0.5962  0.3358 0.4258 0.5423  0.674
 2000.  0.0500  -0.0600   0.2919 0.6495 0.7121  0.3219 0.4756 0.5743  0.405
 2500.  0.0000   0.0000   0.2716 0.8403 0.8831  0.3287 0.5631 0.6520  0.213
 3000. -0.0500   0.0600   0.2592 1.0747 1.1055  0.3452 0.7197 0.7982  0.103
.TE
```

To produce the fancy table shown in table 7, use a UNIX command of this form:

tbl file | troff -mm

TABLE 7. Sample Summary Output After Typesetting

Table ___. Ballistic Errors for a _____ Round Fired from a Stationary _____ with a _____ Range finder and a Computer

			Horizontal (mils)			Vertical (mils)			
Range (m)	Fixed Bias (mils) horiz.	vert.	Ran. error	Var. bias	Total disp.	Ran. error	Var. bias	Total disp.	P_h
500.	0.0000	-0.2100	0.6937	0.3955	0.7985	0.6907	0.4150	0.8058	0.992
1000.	0.0300	-0.1800	0.4130	0.4268	0.5939	0.4130	0.4066	0.5795	0.900
1500.	0.1100	-0.1200	0.3294	0.4969	0.5962	0.3358	0.4258	0.5423	0.674
2000.	0.0500	-0.0600	0.2919	0.6495	0.7121	0.3219	0.4756	0.5743	0.405
2500.	0.0000	0.0000	0.2716	0.8403	0.8831	0.3287	0.5631	0.6520	0.213
3000.	-0.0500	0.0600	0.2592	1.0747	1.1055	0.3452	0.7197	0.7982	0.103

2. MODEL OVERVIEW

TAM finds the super elevation and time of flight to the desired target ranges, and then finds the components of the error budget. This may be thought of as filling two tables, one for horizontal and the other for vertical error components. These two tables are the error budget tables. As they are filled, the components are aggregated into fixed biases, variable biases, and random errors. Finally, TAM finds the hit probability on a standard NATO target and prints the results.

The Tank Accuracy Model (TAM) has the following features:

1. **Modular:** It is divided into coherent routines written in a page or less of code.
2. **Commented:** It was developed using successive refinements. At each step, the pseudo code became comments in the program.
3. **Indented:** It uses indentation to show the relation of the statements to each other.
4. **Structured:** Each routine is written using only simple sequences, simple branches, and simple loops. In no case does a branch or loop have multiple entrances or exits.
5. **Mnemonic:** The modular nature of the program minimizes the number of variables the programmer must comprehend in the context of a single routine and allows the use of simple, mnemonic variable names. The variable names are one or two characters, where the second is a subscript. This allows the documentation to explain the code using clear, compact equations. Greek letters are spelled out in the code and are the only exception for integers and reals. Since logical and character variables generally do not occur in equations, they may be longer than two characters.
6. **Nearly GOTO-less:** The single DO-UNTIL statement occurs in the FIRE routine and is implemented as a clean backwards GOTO. As is widely recommended, TAM does not use arithmetic IF's, assigned GOTO's, or alternate returns.
7. **Emphatic as needed:** TAM uses upper case only to highlight nonsequential flow of control. This includes branching, looping, beginning, and ending of routines. It does not include subroutine or function calls.

Error Budget Tables. A major task of the program is to fill the error budget tables. Understanding just how to calculate the values in these tables is the key to understanding the program. Section 1.3 illustrated the error budget tables. Table 8 shows the organization of the variables that store the error budget for the horizontal errors. The vertical errors are stored in the same fashion but the array containing them is v_{11} to v_{mn} rather than h_{11} to h_{mn}.

TABLE 8. Organization of Error Budget Tables

r_1	r_2	...	r_n
h_{11}	h_{12}	...	h_{1n}
h_{21}	h_{22}	...	h_{2n}
.			
.			
.			
h_{m1}	h_{m2}	...	h_{mn}

2.1 Organization

TAM code is generally organized in a hierarchy, although a few utility routines are called by several branches of the hierarchy. TAM uses labeled common statements to communicate global variables. Both of these are discussed below.

Hierarchy of routines. The main routine of TAM has the following hierarchy of routines under it. Called routines are indented under the routines that call them.

TAM	Produces accuracy and hit probability data.
- DRAG0	Reads drag information and aggregates constants.
- SUPERN	Finds super elevation and time of flight for each range.
- SUPER	Finds super elevation and time of flight for one range.
- FIRE	Finds time of flight given super elevation, etc.
- DRAGF	Finds drag at current velocity.
- FB	Finds fixed bias errors.
- VBH	Finds horizontal variable bias errors and related vertical errors.
- SUPER	(& subsidiary routines)
- VBV	Finds remaining vertical variable bias errors.
- SUPER	(& subsidiary routines)
- RE	Finds random errors.
- PRE	Prints error budget.
- PRTBL	Prints summary table.
- NDTR	Finds integral under standard normal density function.

Global variables. Some variables are used in many routines and are communicated to those routines using labeled common statements. The common statements are:

```
common /atmosp/ ta, rho
common /budget/ h(20,20), v(20,20)
common /output/ table(20,9)
common /ranges/ n, r(20)
common /trajec/ v0, t(20), theta(20)
```

The variables shown above correspond to the clean mathematical variables shown and defined below.

Value	Comment
ρ	Air density (kg/cu m).
θ_i	Super elevation for ith target range (rad).
h_{ij}	Horizontal error budget table (mils).
n	Number of ranges.
r_i	Range to target (m).
t_a	Air temperature ^{0}C.
t_i	Time of flight to ith range (s).
$table_{ij}$	Summary table of errors at ith range for jth type (mils).
v_0	Muzzle velocity (m/s).
v_{ij}	Vertical error budget table (mils).

2.2 Error Messages

You may receive the error messages discussed below.

1. Message: 'Divide by zero.' This message will be produced by the operating system rather than by TAM. Possible reasons are as follows:
   - An input range is zero. Change to a nonzero value.
   - An input velocity is zero. Change to a nonzero value.
   - The ballistic coefficient is zero. Change to a nonzero value.
   - A value lies outside a table provided in input. Use the appropriate table.
   - The range error is equal to a target range. Use a range error that is less than the target range.
   - An error in column 4 or 8 of the summary table is zero. Either change the lay error or the round-to-round dispersion to a value greater than zero.
2. Message: 'Drag0: Line 2 of drag data >20'. The program only allows space to store 20 mach numbers and 20 drag numbers. Solution: Delete some of the number pairs. Alternatively, increase the dimension of the relevant arrays, recompile, and rerun.
3. Message: 'Constant fixed bias makes no sense.' This is just a warning message. The program will run fine. The output will be for a system that no tank designer would make.
4. Message: 'VBH: cant integer must be -1 or -2'. Solution: If the system does not use a cant sensor, use '-1'; if it does, use '-2'.

2.3 Routine TAM - The Main Program

The main routine reads control data and then calls subsidiary routines to generate the super elevation, time of flight, error budget values, hit probabilities and to print

the results. It is short so it can be modified by imbedding the executable statements in loops or whatever the user desires.

The main routine uses the following local variables:

Value	Comment
flag(1)	If true, echo the input to output.
flag(2)	If true, print error budget tables.
flag(3)	If true, print summary data for input to another computer program.
flag(4)	If true, print summary data for input to the Troff typesetter.
whyrun	Line containing reason for run. Documents input & output.
fdate	Function returning current date and time.

```
      PROGRAM TAM
c     Generate tank accuracy data.
      common /budget/ h(20,20), v(20,20)
      common /ranges/ r(20), n
      common /trajec/ v0, t(20), theta(20)
      common /atmosp/ ta, rho
      logical flag(4)
      character whyrun*78, fdate*24
1     format(a)
2     format(f8.3,'',a)
3     format(4i2,a)
4     format(10f8.0)

      read 1, whyrun
      print 1, whyrun
      print 1, fdate()
c     Read control info
        read *, flag
        read *, n
        read *, (r(i),i=1,n)
c     Read atmospheric data & round info
        read *, ta, rho
        read *, v0
        IF (flag(1)) THEN
          print 3, flag,'Print flags'
          print 1, 'TARGET RANGES (m)'
          print 4, (r(i),i=1,n)
          print 2, ta, 'Air temperature (C)'
          print 2, rho, 'Air density (kg/cu m)'
          print 2, v0, 'Muzzle velocity (m/s)'
        ENDIF
        call drag0(0,ta,rho,flag(1))
c     Zero error budget tables.
```

```
      DO 30 i=1,20
        DO 20 j=1,20
          h(i,j) = 0.0
          v(i,j) = 0.0
20      CONTINUE
30    CONTINUE
      call supern(flag(1))
      call fb(flag(1))
      call vbh(flag(1))
      call vbv(flag(1))
      call re(flag(1))
      if (flag(2)) call pre
      call prtbl(flag)
      END
```

INTENTIONALLY LEFT BLANK.

3. TRAJECTORY ROUTINES

The most complicated portion of TAM is the calculation of the appropriate super elevations and times of flight. The routines which make these calculations are SUPERN, SUPER, FIRE, and DRAG0. The very first one executed is DRAG0. It reads in information about the atmosphere and round and calculates a few values which will be used repeatedly later. Next, TAM calls SUPERN to find the super elevations and times of flight to each of the target ranges. SUPERN in turn calls SUPER to find these values for a single target range. SUPER is also called by VBV at a later time to find some of the vertical errors. SUPER calls FIRE several times, each time correcting the super elevation until it 'gets the round on target.' FIRE does the actual calculation of the trajectory. Given a super elevation and other important values, it finds the height of the round when it reaches the target range and also finds the time of flight. In the process of finding the trajectory, FIRE calls the DRAG entry in the DRAG0 routine to return a key drag value.

3.1 Routine SUPERN - Find Super Elevation at Each Range

SUPERN loops through the target ranges and calls the SUPER routine to find the super elevation and time of flight for that target range. The super elevation angle returned is in radians, so it is converted to mils. Then, if the user set the appropriate print level, the target range, super elevation, and time of flight are printed.

SUPERN uses the following local variables:

Value	Comment
echo	If true, print tgt ranges, super elev, and time of flight.
θ_m	Super elevation (mils).

```
      SUBROUTINE SUPERN (echo)
c     Supern: find super elevation at n ranges.
      common /ranges/ r(20), n
      common /trajec/ v0, t(20), theta(20)
      real thetam(20)
      logical echo
1     format(a)
2     format(10f8.0)
3     format(10f8.3)

      DO 20 i=1,n
        call super(v0,r(i),0.0,t(i),theta(i))
        thetam(i)= theta(i)*3200./3.1416
20    CONTINUE
      IF (echo)  THEN
        print 1,
     1'TRAJECTORY: range (m), super elev (mils), time of flight (sec)'
        print 2, (r(i),i=1,n)
        print 3, (thetam(i),i=1,n)
```

```
  print 3, (t(i),i=1,n)
ENDIF
END
```

3.2 Routine SUPER - Find Super Elevation at One Range

SUPER finds the super elevation angle of the gun and the time of flight of the round. This is the elevation angle above the line of sight which compensates for the drop of the round caused by gravity.

SUPER first approximates the super elevation (θ) using a closed form formula[1]. Then SUPER calls the fire routine three times to calculate the trajectory and corrects θ after each trajectory is completed. Each time FIRE is called, it returns z, the vertical error, and t, the time of flight. Three iterations finds θ to sufficient accuracy for our purposes.

Figure 1 illustrates θ, the super elevation. SUPER finds the initial estimate for θ as follows:

Value	Comment
$g = 9.8$	Gravitational acceleration (m/s)
$v_0 = input$	Muzzle velocity (m/s).
$v_w = input$	Range wind speed (m/s).
$d = 120.$	A drag related value (m/s per km)
$c = d/(1000v)$	A drag related value.
r	Target range (m).
$a = g/(4rc^2v^{2)}$	Grouping.
$b = 2cr$	Grouping.
$\theta \approx a(e^b - b - 1)$	Initial estimate of super elevation angle (rad).
z	Height of round at target range (m).
$\theta' = \theta - z/r = output$	Corrected super elevation (rad).
$t = output$	Time of flight (sec).

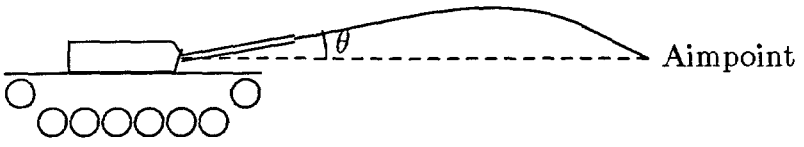


Figure 1. The Super Elevation Angle θ

1. Johnson, L. D., *Engagement Range Bracketing for Multiple Battlesights in Tank Gunnery,* ARBRL-TR-02362, Sep 1981, Ballistic Research Laboratory, Aberdeen Proving Ground, MD, p 12.

```
      SUBROUTINE SUPER(v0,r,vw,t,theta)
c     Super: find super elevation.
      real k

c     Make first estimate of super-elevation.
        d = 120.
        k = d/(1000.*v0)
        a = 9.8/(4*r*(k*v0)**2)
        b = 2.0*k*r
        theta = a*(exp(b) - b - 1.0)
      DO 10 j=1,4
        call fire(theta,v0,r,vw,z,t)
        theta = theta - z/r
10    CONTINUE
      END
```

3.3 Routine FIRE - Find the Trajectory of a Point Mass

Fire solves the differential equations of motion to determine the trajectory of a bullet fired at a target. It requires launch angle, muzzle velocity, target range, and drag as input.

The Firing Tables Branch of ARDEC generates drag data that is easily converted to the form used here. See the next section for more details about drag data.

Finding the velocity of direct fire rounds is easy, since there is little change in air density for a flat trajectory and since the time of flight is so short that factors like the Earth's rotation can be ignored.

Given the position x_0, and velocity $\dot{x}_0$ at time zero, and the relationship $\ddot{x} = -kv\dot{x}$ to find the acceleration, then the position at time t is:

$$x = x_0 + \dot{x}_0 t + \ddot{x}_0 t^2/2$$

But this is only exact if the acceleration is constant! It is not; it is changing monotonically in the region of interest, so let's use $\ddot{x}_i = (\ddot{x}_0 + \ddot{x})/2$ as a better approximation to the average acceleration during the interval.

Value	Comment
	Arguments:
$\theta = input$	Super elevation (rad).
$v_0 = input$	Muzzle velocity (m/s).
$r = input$	Target range (m).
$v_w = input$	Range wind (m/s).

$output = y$	Vertical coordinate of round at target range (m).
$output = t$	Time of flight (sec).
	Initial values:
$x = 0$	Horizontal coordinate of round at $t = 0$
$y = 0$	Vertical coordinate of round at $t = 0$
$\Delta t = 0.002$	Time increments (sec).
$t = 0$	Initial time (sec).
$\dot{x} = v_0 \cos\theta$	Initial horizontal velocity (m/s).
$\dot{y} = v_0 \cos\theta$	Initial vertical velocity (m/s).
$v = \sqrt{\left(\left(\dot{x} - v_w\right)^2 + \dot{y}^2\right)}$	Apparent velocity (includes range wind) (m/s).
	Iterated values:
$k = f(v)$	Drag multiplier.
$\ddot{x}_0 = -kv\dot{x}_0$	Acceleration at beginning of interval.
$\dot{x} = \dot{x}_0 + \ddot{x}_0 \Delta t$	Velocity at end of interval.
$\ddot{x}_i = (\ddot{x}_0 + \ddot{x})/2 = (\ddot{x}_0 - kv\dot{x})/2$	Acceleration at mid-point of interval.
$x = x_0 + \dot{x}_0 \Delta t + \ddot{x}_i \Delta t^2/2$	Position at end of interval.
$\ddot{y}_0 = -kv\dot{y}_0$	Acceleration at beginning of interval.
$\dot{y} = \dot{y}_0 + \ddot{y}_0 \Delta t$	Velocity at end of interval.
$\ddot{y}_i = (\ddot{y}_0 + \ddot{y})/2 = (\ddot{y}_0 - kv\dot{y})/2$	Acceleration at mid-point of interval.
$y = y_0 + \dot{y}_0 \Delta t + \ddot{y}_i \Delta t^2/2$	Position at end of interval.
	Termination values:
$v = \sqrt{\left(\left(\dot{x} - v_w\right)^2 + \dot{y}^2\right)}$	Apparent velocity (includes range wind) (m/s).
$t = t + 2\Delta t$	Increment time (sec).
$r_g = \sqrt{(x^2 + y^2)}$	Distance to round (m).
$t_r = 0.5(r - r_g)/v$	Half the time remaining (sec).

```
      SUBROUTINE FIRE(theta,v0,r,vw,y,t)
c     Find trajectory info.
      real k
      data g /9.80665/

c     Find initial conditions
        x = 0
        y = 0
        dt = .002
        t = 0.
        dx = v0*cos(theta)
        dy = v0*sin(theta)
        v = sqrt((dx-vw)**2 + dy**2)
```

```
c     Find position at current time
20    CONTINUE
        call dragf(k,v)
        dt2 = 2.*dt
        ddx = -v*k*(dx-vw)
        dx3 = dx +ddx*dt2
        ddx = .5*(ddx-v*k*(dx3-vw))
        dx2 = dx + ddx*dt
        x   = x + dx2*dt2
        dx = dx3

        ddy = -v*k*dy - g
        dy3 = dy + ddy*dt2
        ddy = .5*(ddy-v*k*dy3-g)
        dy2 = dy + ddy*dt
        y   = y + dy2*dt2
        dy  = dy3

        v  = sqrt((dx-vw)**2 + dy*dy)
        t  = t+dt2
        rg = sqrt(x*x+y*y)
        tr = 0.5*(r - rg)/v
        if (dt .gt. tr) dt = tr
      IF (rg.lt.r-.0001) GOTO 20
      END
```

3.4 Routine DRAG0 - Find the Drag Force on the Round

DRAG0 finds the coefficient k for these equations which give the deceleration of a bullet:

$$\ddot{x} = -kv\dot{x}$$
$$\ddot{y} = -kv\dot{y} - g$$

The equation for k is given below. Some of the values in the equation will not change and are computed once at the beginning of a TAM run. The air temperature and air density change a few times during a TAM run. The drag coefficient changes with every time step in the FIRE routine.

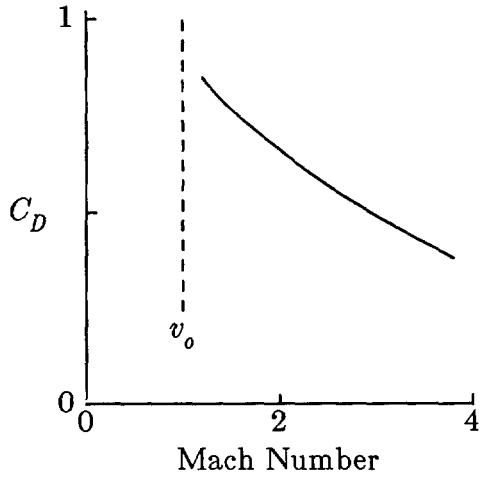
$$k = C_D \pi \rho / 8 b_c$$

TAM calls DRAG0 once at the beginning of each run with the argument j set to zero. This causes DRAG0 to read the ballistic coefficient and drag coefficient table. Later TAM calls DRAG0 with new values for air temperature or air density but the argument j is set to a non-zero value. This causes DRAG0 to skip the reading of the ballistic coefficient and drag coefficient because those values have already been read in.

The ballistic coefficient is simply the flight mass divided by the cross sectional area. That area is simply $\pi d^2/4$, where d is called the reference diameter. TAM expects the

ballistic coefficient in pounds per square inch and converts it to kilograms per square meter for internal use. (The ballistic coefficient is usually stated in English units and it's easier to let the program do the conversion.)

DRAG0 reads the 'drag curve' or ballistic coefficients as a table. Figure 2 illustrates both the curve and the table. DRAGF finds the current mach number and linearly interpolates in the table to find the drag coefficient. It then multiplies by c_1, which is an aggregation of values that stay constant during the flight of the projectile.



Mach #	C_D
1.2	.86
1.4	.80
1.8	.71
2.5	.57
3.0	.50
3.8	.39

Figure 2. Drag Coefficient for a Typical 105mm HEAT Round.

The International Civil Aviation Organization (ICAO) standard atmosphere describes the mean conditions of the atmosphere at sea level at freezing temperature. Use the standard atmosphere at sea level to find the speed of sound v_t at the desired temperature t in degrees centigrade. A temperature of 15 degrees centigrade (59 degrees Fahrenheit) is customarily used for the temperature at which combat occurs.

Value	Comment
	Constants:
$t_0 = 273.15$	Absolute air temperature at freezing (Kelvin).
v_0=331.3	Speed of sound (m/s) (freezing and sea level).
	Inputs:
$j = input$	If zero, read data.
$t = input$	Air temperature (deg C).
$\rho = input$	Air density (kg/m^3).
$echo = input$	If true, echo input.

Value	Comment
	Read and calculate:
$b_c = 0.4535924 \cdot 39.37^2 b_c'$	Convert ballistic coefficient (kg/m^2).
n	Number of drag coefficient values in table.
x_i	Mach numbers.
c_i	Drag coefficients.
v	Velocity of round (m/s).
$v_s = v_0\sqrt{(t_0+t)/t_0}$	Speed of sound (m/s) at current temperature and pressure.
$c_1 = \pi\rho/8b_c$	Aggregation of constants.
	Find these at each time step:
$m = v/v_s$	Mach number.
$C_D = c_i+(c_{i+1}-c_i)(m-x_i)/(x_{i+1}-x_i)$	Drag coefficient.
$k = C_D c_1$	Drag multiplier.

```
      SUBROUTINE DRAG0(j,t,rho,echo)
c     Read and find drag related data.
      real x(20), c(20), k, m
      logical echo
      data t0, v0 /273.15, 331.3/
      save
1     format(a)
2     format(10f8.3)
3     format(f8.3,'',a)

      IF (j.eq.0) THEN
c       Read ballistic coefficient in lbs/in**2
          read *, bc
          if (echo) print 3, bc, 'Ballistic coefficient (lbs/sq in)'
c       Convert to kg/m**2
          bc = bc*.4535924*(39.37)**2
        read *, n
        if (n.gt.20) print *,'Drag0: Line 2 of drag data >20'
        IF (n.gt.20) STOP
        read *, (x(i), c(i), i=1,n)
        IF (echo) THEN
          print 1,'DRAG DATA: Mach nr, KD'
          print 2, (x(i),i=1,n)
          print 2, (c(i),i=1,n)
        ENDIF
      ENDIF
      vs = v0*sqrt((t0+t)/t0)
      c1 = 3.1416*rho/(8.0*bc)
      RETURN
```

```
      ENTRY DRAGF (k,v)
c     Drag: Find drag factor k.

      m = v/vs
      call hunt (x,n,v/vs,i)
      cd = c(i)+(c(i+1)-c(i))*(m-x(i))/(x(i+1)-x(i))
      k = cd*c1
      END
```

4. ERROR BUDGET ROUTINES

The next group of routines find fixed biases, variable biases and random errors.

4.1 Routine FB - Fixed Bias Calculations

Fixed biases are permanent errors. For example, the M60 tank does not correct for parallax or the drift of a spinning round. The M1 tank corrects for these so it has zero fixed biases. Very often, fixed biases are computed from firing data and while the firing data indicates near zero fixed biases for the M1, it is not clear that it is justified in setting them to zero. Another way for modern tanks to experience fixed biases is if the laser range finder is broken or the commander issues orders to use battle sight ranging. This speeds the firing process but introduces some fixed biases.

Row 1: Find fixed bias directly. Often, you will want to set the fixed bias to zero, perhaps to a constant, or interpolate in a table. If m is zero or greater, FB will call RDTBL to handle this. See the section discussing RDTBL for more details. FB places the resulting values on line 1 of the error budget tables. There is probably no reason to set the fixed bias to a constant, of course.

Rows 1-3: Find fixed bias components. The alternative to finding the fixed bias directly is to calculate the parallax errors (row 2), the drift errors (row 3), or both. Then sum the values to find the fixed biases (row 1). If the fire control system does not correct for drift and parallax, then the fixed biases must be computed.

There is one complication. It's called battle sight ranging. It affects both parallax and drift. If the fire control computer uses bad range data to correct for parallax or drift, the fixed biases will change. The range data will be bad if the tank uses battlesight ranging. Why would it use battlesight ranging?

When tanks such as the M60A1 used coincident range finders, it might take six or so seconds to get a proper range to the target. To reduce this time, a preset range might be set into the fire control system. Later, when the gun was actually fired at the target, this range might be correct or might not. In any case, the parallax must be computed.

The tank will also use battlesight ranging if the laser range finder is broken. In this case, the gunner feeds a range estimate into the computer.

At this point, let's define some variables. Most of them will be used for calculating both parallax and drift.

Value	Comment
c_m	Conversion from radians to mils.
r_i	Range to the target (m).
r_e	Expected range of the target (m).
h_g, v_g	Horizontal and vertical offset of gun from sight (m).
h', v'	Horizontal and vertical position of projectile at target (m).
h, v	Horizontal and vertical position of projectile at target (mils).

Row 2: Find parallax. Tank guns are always offset from the sight. If the fire control does not correct for this or corrects based on inaccurate range information, horizontal and vertical errors occur. The fire control can treat parallax three ways: a) ignore it, b) correct for it using battle sight ranging, or c) correct for it using accurate ranging.

If the fire control does not correct for this offset, there will be an error except at the zeroing range. If the fire control corrects based on the battle sight range, there will be an error except at the battle sight range. Any deviation from the expected range causes an error. The program reads the expected range, then the horizontal and vertical offsets of the gun from the sight.

Figure 3 illustrates horizontal parallax error.

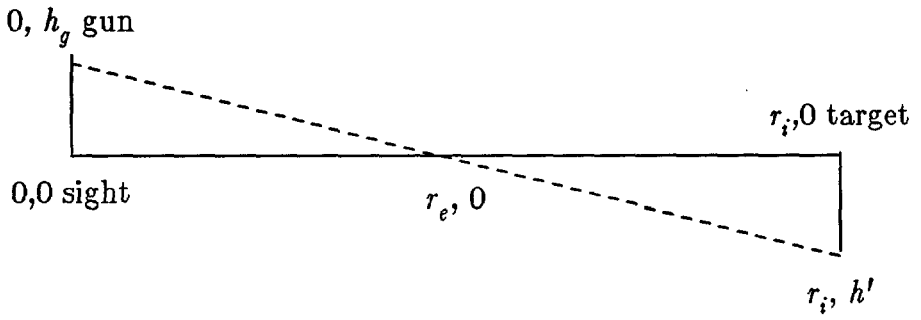


Figure 3. Errors Due to Parallax

FB reads three values: the expected range of the target (r_e) and the horizontal and vertical offsets (h_g, v_g) of the gun from the sight. If the system corrects for parallax, the value for the expected range should be zero, in which case, FB leaves the parallax at zero.

The derivation of the error equation is:

Value	Comment
$h'/(r_i - r_e) = h_g/r_e$	By similar triangles.
$h' = h_g(r_i - r_e)/r_e$	Solving for linear error (m).
$h_{2i} = c_m h'/r_i = c_m h_g(r_i - r_e)/(r_e r_i)$	Converting to angular error (mils).
$v_{2i} = c_m v_g(r_i - r_e)/(r_e r_i)$	And similarly for the vertical.

Row 3: Find drift. If a round is spinning, it will curve (drift) to the right or left depending on the direction of spin. The reasons for drift are beyond the scope of this report.

Figure 4 illustrates drift. If the spinning round is launched 'directly at the target,' it will curve horizontally and pass throught some point A. Normally, the gun will be adjusted so that it points in a slightly different direction such that the round strikes the target at some expected range r_e. The second curve passing through the line of sight at r_e and ending at B illustrates this adjustment. This adjustment occurs when the gun is zeroed or when the fire control corrects for battle sight ranging. If the fire control makes

no other correction, the round strikes right of the aim point on a target closer than the expected range and left of the aim point on a target beyond the expected range.

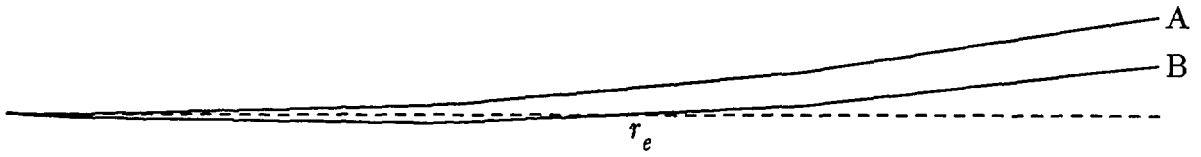


Figure 4. Error Due to Drift

The final case, not shown, is when the fire control corrects for drift with accurate range information. In this case, the drift error will be zero.

The error due to drift is:

Value	Comment
	Case 1: Unzeroed.
$q = t$	For independent variable time of flight (sec).
$q = c_m \tan s_e$	For independent variable super elevation (mils).
$q = r/1000$	For independent variable target range (m).
$f(q) = \sum_{i=1}^{5} d_i q^{i-1}$	Linear drift via polynomial fit.
	Angular drift at expected range:
$h_e = c_m f(t_e)/r_e$	If independent var is time of flight (k=1)
$h_e = c_m f(c_3 \tan s_e)/r_e$	If independent var is super elevation (k=2)
$h_e = c_m f(r_e/1000)/r_e$	If independend var is range (k=3)
	Drift at actual range less correction:
$h_{i3} = c_m f(t_i)/r_i - h_e$	If independent var is time of flight (k=1)
$h_{i3} = c_m f(c_3 \tan s_i)/r_i - h_e$	If independent var is super elevation (k=2)
$h_{i3} = c_m f(r_i/1000)/r_i - h_e$	If independent var is range (k=3)

```
      SUBROUTINE FB (echo)
c     Find fixed biases.
      common /budget/ h(20,20), v(20,20)
      common /ranges/ r(20), n
      common /trajec/ v0, t(20), theta(20)
      character*20 str
      logical echo
      real d(5)
      f(q) = d(1)+ q*(d(2)+ q*(d(3)+ q*(d(4)+ q*d(5))))
1     format(f8.3,'',a)

      read *, m
      IF (m.gt.1) THEN
c     Row 1: Find fixed biases directly, ignoring components.
```

```
        str = 'FIXED BIAS DATA'
        call rdtbl(1,m,str,echo)
      ELSEIF (m.eq.1) THEN
        print *, 'Constant fixed bias makes no sense.'
      ELSEIF (m.lt.0) THEN
c     Rows 1-3: Find fixed bias components, then sum.
        cm = 3200./3.1415926535
c       Row 2: Find parallax error.
          read *, re, hg, vg
          if(echo)print 1, re,'Expected range (zeroing or battle sight) (m).'
          if(echo)print 1, hg,'Horizontal offset of gun from sight (m)'
          if(echo)print 1, vg,'Vertical offset of gun from sight (m)'
          IF (hg.ne.0.0 .or. vg.ne.0.0) THEN
            DO 20 i=1,n
              h(i,2) = cm*hg*(r(i)-re)/(re*r(i))
              v(i,2) = cm*vg*(r(i)-re)/(re*r(i))
20          CONTINUE
          ENDIF
c       Row 3: Find drift error.
          read *, k
          if(echo)print *, k, 'ID of drift equation.'
          IF (k.gt.0) THEN
            read *, d
            if(echo)print *, 'Drift coefficients are:'
            if(echo)print *, d
c           Find drift at expected range.
              if (k.eq.1 .or. k.eq.2) call super(v0,re,0.0,te,se)
              if (k.eq.1) he = cm*f(te)/re
              if (k.eq.2) he = cm*f(tan(se))/re
              if (k.eq.3) he = cm*f(re/1000)/re
            DO 30 i=1,n
              if (k.eq.1) h(i,3) = cm*f(t(i))/r(i) - he
c             if (k.eq.2) h(i,3) = cm*f(tan(theta(i)))/r(i) - he
              if (k.eq.3) h(i,3) = cm*f(r(i)/1000.0)/r(i) - he
30          CONTINUE
          ENDIF
c       Row 1: Sum all fixed bias errors.
          DO 40 i=1,n
            h(i,1) = h(i,2) + h(i,3)
            v(i,1) = v(i,2)
40        CONTINUE
      ENDIF
      END
```

4.2 Routine VBH - Horizontal Variable Bias Calculations

VBH finds the horizontal variable biases and related vertical biases. These biases are caused by cant, crosswind, jump, fire control implementation, boresight retention, and parallax / drift compensation.

Row 5: Cant error. When the firing platform is canted and the cant is unmeasured or imperfectly measured, a delivery error occurs. The left half of Figure 5 illustrates cant error when cant is not measured. Assume the target is at the origin (at the tails of the two arrows representing the coordinate axes). The fire control will elevate the gun through the super elevation angle θ_i which corresponds to a distance in the target plane. If the firing platform is canted, the extension of the gun barrel traces the circle above the target. Gravity drop will cause the round to fall and strike a point on the lower curve directly below where the gun is pointing on the upper curve.

The right half of Figure 5 illustrates cant error when a cant sensor measures but does so imperfectly. In this case, the cant of the firing platform is measured to be ϕ, but it is actually $\phi+\omega$. Thus the uncorrected error is h_i, v_i.

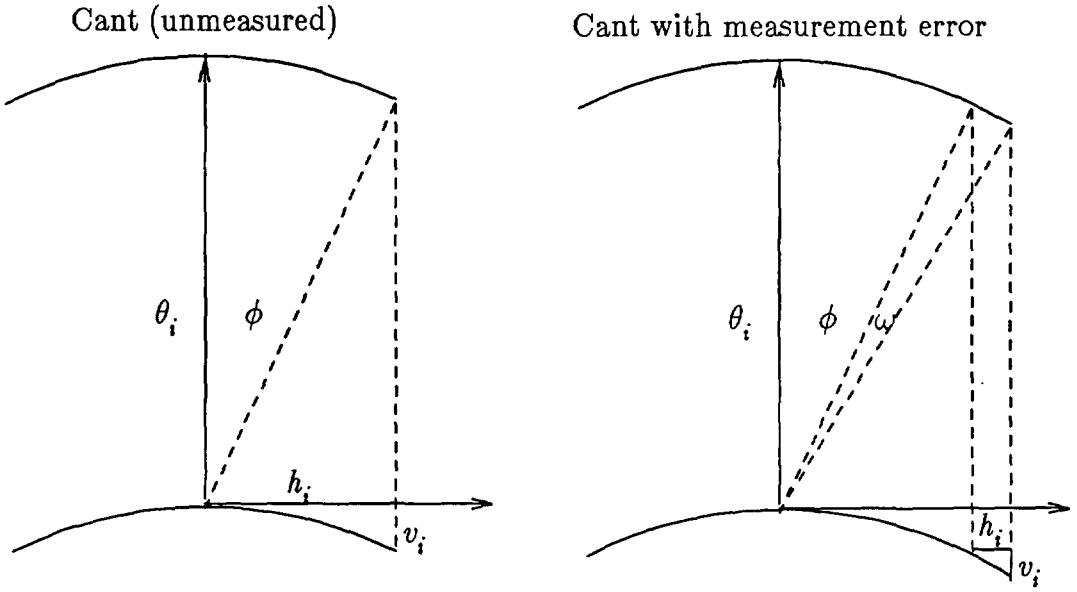


Figure 5. Errors Due to Cant

Value	Comment
c_m	Mils/radian
c_r	Radians/degree
θ_i	Super elevation for range r_i (rad)
	For unmeasured cant:
ϕ	Std. dev. of cant (rad)
$h_{i,5} = c_m\theta_i\sin\phi$	Horizontal cant error (mils)
$v_{i,15} = c_m\theta_i(1-\cos\phi)$	Vertical cant error (mils)

	For imperfectly measured cant:
ϕ	Mean of absolute value of cant (rad)
ω	Std. dev. of cant measurement error (rad)
$h_{i,5} = c_m \theta_i \sin(\omega)$	Horizontal cant error (mils)
$v_{i,15} = c_m \theta_i (\cos(\omega) - \cos(\phi + \omega))$	Vertical cant error (mils)

Row 6: Crosswind error. When an unmeasured crosswind exists, or the crosswind measurement is in error, the round will be deflected horizontally. This error persists during the firing of more than one round but on average, the mean of the error is zero. For this reason, it is a variable bias. TAM uses the 'classical' crosswind formula[2] to find the angular error due to crosswind or crosswind measurement error.

Value	Comment
c_m	Mils/radian
r_i	Target range (m)
t_i	Time of flight (s)
v_o	Muzzle velocity (m/s)
v_w	Crosswind velocity error (m/s)
$e = v_w(t_i - r_i/v_o)$	Linear error (m) (Classical cross wind formula)
$h_{i,6} = c_m e / r_i$	Angular error (mils)

Row 7: Jump error. Jump is all unexplained systematic error that persists for more than one round. Its mean is zero, so it is a variable bias. Schmidt[3] at ARL has done some research that suggests it is due to balloting of the round as it travels down an imperfectly smooth barrel. Normally, jump is a function of range and AMSAA generates the data when they analyze test firings. Since it is based on field data and is a function of target range, the data is read in as a table.

$$h_{i,7} = \text{interpolated value in RDTBL}$$
$$v_{i,7} = \text{interpolated value in RDTBL}$$

Row 8: Fire control error. This error is due to round off error and other imperfections in the computation of the fire control solution as well as errors in pointing the gun in the exact direction specified by the fire control computer. It is a function of range and is read in as a table. You can get the data from AMSAA.

$$h_{i,8} = \text{interpolated value in RDTBL}$$

2. Dick Norman, Letter to President, US Army Armor and Engineering Board, with regard to Formulas Being Used to Compute Errors and Biases HPC as Compared With Formulas in R-1380 and R-1937, 22 Oct 79. (This in turn references Frankford Arsenal R-1380A.)
3. Unpublished work at ARL.

$v_{i,8}$ = interpolated value in RDTBL

Row 9: Boresight error. This error, caused when boresighting the gun, is read in as a pair of constants, and is obtainable from AMSAA.

$$h_{i,9} = h_e$$
$$v_{i,9} = v_e$$

Row 10: Parallax, drift compensation. When a modern, fully functional tank corrects for parallax and drift, the computation assumes the range to the target is known perfectly. If this is not true, the gun will be aimed slightly in error.

Value	Comment
d	Target range error (m)
c_m	Mils per radian
r_i	Target range (m)
h_g	Horizontal distance from gun to sight (m)
θ_1	Super elevation at range $r_i - d$ (rad)
θ_2	Super elevation at range $r_i + d$ (rad)
$d_2 = h_g d/(r_i^2 - d^2)$	Parallax, range error (rad)
$d_3 = \theta_1 - \theta_2$	Drift, range error (rad)
$h_{i,10} = c_m \lvert d_2 + .5 d_3 \rvert$	Parallax / drift compensation error (mils)

```
      SUBROUTINE VBH(echo)
c     Find horiz variable biases, related vert ones.
      common /ranges/ r(20), n
      common /budget/ h(20,20), v(20,20)
      common /trajec/ v0, t(20), theta(20)
      common /atmosp/ ta, rho
      character str*20
      logical echo, finned
1     format(f8.3,'',a)
2     format(a)

      cm = 3200/3.1416
      cr = 3.1416/180.
c     Row 5: Find error due to cant.
        read *, m
        if (m.eq.-1) read *, phi
        if (m.eq.-2) read *, phi, omega
        if (echo) print 1, phi, 'Std. dev. of cant (deg)'
        if (echo.and.m.eq.-2)
     1    print 1, omega, 'Std. dev. of cant sensor (deg)'
        omega = omega*cr
        phi = phi*cr
        DO 10 i=1,n
```

```
          IF (m.eq.-1) THEN
            h(i,5) = cm*sin(phi)*theta(i)
            v(i,15) = cm*theta(i)*(1.0-cos(phi))
          ELSEIF (m.eq.-2) THEN
            h(i,5) = cm*sin(omega)*theta(i)
            v(i,15) = cm*theta(i)*(cos(phi)-cos(phi+omega))
          ELSE
            print*,'VBH: cant integer must be -1 or -2'
          ENDIF
10      CONTINUE
c     Row 6: Find error due to crosswind.
        read *, vc
        if (echo) print 1, vc, 'Cross wind (m/s)'
        DO 20 i=1,n
          h(i,6) = cm * vc * (t(i)-r(i)/v0) / r(i)
20      CONTINUE
c     Row 7: Find error due to jump.
        read *, m
        str = 'JUMP VALUES'
        call rdtbl(7,m,str,echo)
c     Row 8: Find error due to fire control system
        read *, m
        str = 'FIRE CONTROL VALUES'
        call rdtbl(8,m,str,echo)
c     Row 9: find error due to boresight retention.
        read *, he, ve
        if (echo) print 1, he, 'Horizontal boresight value (mils)'
        if (echo) print 1, ve, 'Vertical boresight value (mils)'
        DO 30 i=1,n
          h(i,9) = he
          v(i,9) = ve
30      CONTINUE
c     Row 10: Find error due to parallax, drift compensation.
        read *, finned, d, hg
        DO 35 i=1,n
          d2 = -cm*hg*d/(r(i)**2-d**2)
          IF (finned) THEN
            theta1 = 0
            theta2 = 0
          ELSE
            call super(v0,r(i)-d,0.,dm,theta1)
            call super(v0,r(i)+d,0.,dm,theta2)
          ENDIF
          h(i,10) = abs(d2+.5*cm*(theta1-theta2))
35      CONTINUE
c     Row 4: Find total vertical variable bias
        DO 90 i=1,n
```

```
            ss = 0.0
            DO 89 j=5,12
              ss = ss+ h(i,j)**2
89          CONTINUE
            h(i,4) = sqrt(ss)
90        CONTINUE
        END
```

4.3 Routine VBV - Vertical Variable Bias Calculations

VBV finds the remaining vertical variable biases not found in the VBH routine.

Row 5: Muzzle velocity variation. The program reads a value for the standard deviation of muzzle velocity and runs a trajectory with the muzzle velocity increased by one sigma. The error is the difference between the original super elevation and the super elevation for the increased velocity.

Row 6: Find error in PDC due to range error. Parallax and drift compensation computed in the fixed bias routine will be slightly in error if the range measurement is in error.

Row 10: Find vertical error due to range wind. A round traveling into a head wind will strike low and one traveling with a tail wind will strike high.

Row 11: Find vertical error due to air temperature. When the air temperature increases, consequently, the speed of sound increases and the mach number of a round decreases. This increases the drag coefficient C_d. (An air temperature increase also decreases the air density.)

Row 12: Find vertical error due to air density. The gunner enters atmospheric temperature and barometric pressure into the fire control computer from time to time. (During Desert Storm, armor units typically entered this information three times a day according to SFC Holmes of ARL. They only received meteorological data this often.) Because the data may be slightly in error, the gunner may not enter it exactly, or it may vary between settings, a vertical variable occurs.

The VBV routine reads in the standard deviation of the error as a percentage of the nominal value. Then it decrements the barometric pressure by this amount and reruns the SUPER routine to find the appropriate super elevation for the new barometric pressure. The difference between the old and new super elevations is the error due to barometric pressure variations.

Row 13: Find vertical error due to windage jump. When a spinning round leaves the barrel and cross wind is present, the cross wind imparts a force on the spinning round causing it to twist in the vertical plane. This results in a vertical error. For spinning rounds, use a non-zero value (mils). For smooth bore cannon, use a zero error. This routine simply reads in the value (mils) as a component of the vertical variable bias and later root sum squares it to other vertical variable bias terms.

Row 14: Find vertical error due to optical path bending. The routine reads in a constant for this error source and inserts this value in all columns of row 14. It is, of course, an angular error.

```
      SUBROUTINE VBV(echo)
c     Find remaining variable biases.
      common /ranges/ r(20), n
      common /budget/ h(20,20), v(20,20)
      common /trajec/ v0, t(20), theta(20)
      common /atmosp/ ta, rho
      character u*1
      logical echo
1     format(f8.3,'',a)

      cm = 3200/3.1416
c     Row 5: Find error due to muzzle velocity variation.
        read *, f
        if (echo) print 1, f, 'Muzzle velocity variation (m/s)'
        DO 41 i=1,n
          call super(v0+f,r(i),0.,dm,theta1)
          v(i,5) = (theta(i)-theta1)*cm
41      CONTINUE
c     Row 6: Find error due to range estimation + PDC.
        read *, yr, u, vg
        IF (echo) THEN
          if (u.eq.'%') print 1, yr, 'Range Estimation (%)'
          if (u.eq.'m') print 1, yr, 'Range Estimation (m)'
          print 1, vg, 'Vertical gun / sight offset (m)'
        ENDIF
        p = yr
        DO 40 i=1,n
          if (u.eq.'%') p = 0.01*yr*r(i)
          r2=r(i)-100.
          call super(v0,r2,0.,dm,theta1)
          r2=r(i)+100.
          call super(v0,r2,0.,dm,theta2)
          e = p*cm*(theta2-theta1)/200.0
          top     = abs(cm * vg * p )
          bot     = r(i) ** 2  -  p ** 2
          tot     = top/bot
          if (vg.lt.0) tot = -tot
          v(i,6) = abs(e+tot)
40      CONTINUE
c     Row 10: Find error due to range wind.
        read *, x
        if (echo) print 1, x, 'Range wind value (m/s)'
        DO 30 i=1,n
```

```
        call super(v0,r(i),x,dm,theta1)
        v(i,10) = abs(theta(i)-theta1)*cm
30    CONTINUE
c   Row 11: Find error due to air temp
      read *, f
      if (echo) print 1, f, 'Air temperature value (deg C)'
      rho2 = rho*(1-.01*f/2.777777)
      DO 50 i=1,n
        call drag0(1,ta+f,rho,echo)
        call super(v0,r(i),0.,dm,theta1)
        call drag0(1,ta,rho2,echo)
        call super(v0,r(i),0.,dm,theta2)
        v(i,11) = (2.*theta(i)-theta1-theta2)*cm
50    CONTINUE
c   Row 12: Find error due to air density.
      read *, f
      if (echo) print 1, f, 'Air density (fraction reduced)'
      call drag0(1,ta,rho*(1-f),echo)
      DO 60 i=1,n
        call super(v0,r(i),0.,dm,theta1)
        v(i,12) = (theta(i)-theta1)*cm
60    CONTINUE
c   Rows 13, 14: Find error due to windage jump & optical path bending.
      read *, zw
      if (echo) print 1, zw, 'Windage jump (mils)'
      read *, zo
      if (echo) print 1, zo, 'Optical path bending (mils)'
      DO 80 i=1,n
        v(i,13) = zw
        v(i,14) = zo
80    CONTINUE
c   Row 4: Find total vertical variable bias
      DO 90 i=1,n
        ss = 0.0
        DO 89 j=5,14
          ss = ss+ v(i,j)**2
89      CONTINUE
        v(i,4) = sqrt(ss)
90    CONTINUE
    END
```

4.4 Routine RE - Random Error Calculations

RE finds the errors that vary from round to round. They are the dispersion of the round and the lay error. The latter occurs if the gunner attempts to place the cross hairs on the center of mass of the target before each shot; he will not place it in exactly the same spot each time.

The RE routine uses the following variables:

Value	Comment
a, b	Lay error constants.
c_m	Mils per radian.
i	Index of current range.
m	Number of values read in for dispersion.
n	Number of values to print out for dispersion.
r_i	Range to target (m).
h_a, v_a	Horizontal, vertical lay error modifier (mils).
$h_{1,18}$, $v_{1,18}$	Horizontal, vertical round to round error (mils).
$h_{1,19}$, $v_{1,19}$	Horizontal, vertical dispersion (mils).
$h_{1,20}$, $v_{1,20}$	Horizontal, vertical lay error (mils).
h_l, v_l	Horizontal, vertical lay error (mils).

Row 20: Lay error. Lay error is caused by the gunner's inability to lay the sight cross hairs exactly on the center of the target. The amount of lay error depends on the type of fire control. Table 9 below lists the five different ways TAM can handle lay error. This office always uses the quasi-combat numbers, but AMSAA has uses for the others.

TABLE 9. Lay Error Models

TYPE	MATH	COMMENT
0	$h_{i,20} = 300c_m/r_i + .05$	qc - Quasi-combat
1	$h_{i,20} = 150c_m/r_i + .05$	st - Service test
2	$h_{i,20} = 0$	it - Ideal test
3	$h_{i,20} = \sqrt{[(300c_m/r_i)^2 + h_a^2]}$	add-on - What if option
4	$h_{i,20} = h_a$	substitute - What if option

The general form is:

$$h_{i,20} = \sqrt{[(ac_m/r_i + b)^2 + h_a^2]}$$

$$v_{i,20} = \sqrt{[(ac_m/r_i + b)^2 + v_a^2]}$$

Row 19: Round to round dispersion. The dispersion may be treated as a constant or as a function of range. RE calls RDTBL to handle either type. If the dispersion is a function of range, the program interpolates in a table.

Row 18: Random error. The random error is the root sum square of the lay error and the round to round dispersion.

$$h_{i,18} = \sqrt{(h_{i,19}^2 + h_{i,20}^2)},\ v_{i,18} = \sqrt{(v_{19}^2 + v_{i,20}^2)}$$

```
      SUBROUTINE RE(echo)
c     RE: Find random error.
      common /ranges/ r(20), n
      common /budget/ h(20,20), v(20,20)
      character*20 str
      integer n
      logical echo
1     format(a)
2     format(4f8.3,'',a)

      cm = 3200.0/3.1416
c     Read lay error
        read *, a, b, ha, va
        IF (echo) THEN
          print 1,'RANDOM ERROR VALUES'
          print 2, a, b , ha, va, 'Lay error constants'
        ENDIF
      DO 20 i=1,n
c     Row 20: Find lay error.
        hl = a*cm/r(i) + b
        vl = hl
        if (ha .gt. 0.0) hl = sqrt(hl**2 + ha**2)
        if (va .gt. 0.0) vl = sqrt(vl**2 + va**2)
        h(i,20) = hl
        v(i,20) = vl
20    CONTINUE
c     Row 19: Find dispersion.
        read*, m
        str = 'DISPERSION VALUES'
        call rdtbl(19,m,str,echo)
        if (echo) print *
      DO 30 i=1,n
c     Row 18: Find random error by root sum squaring.
        h(i,18) = sqrt(h(i,19)**2 + h(i,20)**2)
        v(i,18) = sqrt(v(i,19)**2 + v(i,20)**2)
30    CONTINUE
      END
```

INTENTIONALLY LEFT BLANK.

5. MISCELLANEOUS ROUTINES

The remaining routines; NDTR, PRE, PRTBL, and RDTBL support routines previously discussed. NDTR is a standard routine for integrating from minus infinity to some value x under the standard normal density function. It is used here to find hit probabilities. PRE prints the error budget table. PRTBL fills the summary table and prints it in plain or fancy form or both. In filling the summary table, it find the hit probabilities for the ninth column of the summary table. RDTBL reads error budget values which don't have to be 'calculated.' It sets a line in the error budget tables to zero, or a constant or interpolates in the values from a table it reads in. The reason for interpolation is that the set of target ranges the user wants to produce usually differ from the target ranges read in as a table.

5.1 Routine NDTR - Integrate the Normal Distribution

NDTR finds the integral under the standard normal density function from $-\infty$ to x using an approximation technique adapted from the NDTR routine in the IBM Scientific Subroutine Package[4]. A simple transformation of the argument makes it usable for normal density functions with $\mu \neq 0$ and $\sigma \neq 1$.

The standard normal density function, shown in Figure 6, is defined by:

$$p = \frac{1}{\sqrt{2\pi}} \int e^{-x^2/2} dx$$

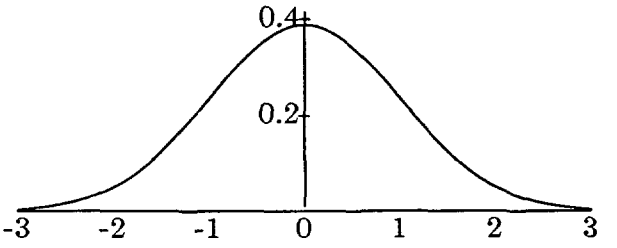


Figure 6. The Area Under the Standard Normal Density Function

Input/Output. This function finds the probability that a normally distributed number is less than x. Thus, the user passes a real argument x and the routine produces an output between 0 and 1. For the standard normal density function, $\mu = 0$ and $\sigma = 1$. If you wish to find the area under a normal density function with any other mean and standard deviation, just use $(x-\mu)/\sigma$ instead of x for the argument.

Mathematics. NDTR uses this approximation by Hastings[5]. (Because of a change of variable, all constants given by Hastings must be multiplied by $\sqrt{2}$).

---

4. System/360 Scientific Subroutine Package (360A-CM-03X) Version III Programmer's Manual, H20-0205-3, IBM Data Processing Division, 112 East Post Road, White Plains, NY 10601, undated, p78.
5. Approximations for Digital Computers, C. Hastings, Jr., Princeton Univ. Press, Princeton, NJ, 1955, p 169.

Various authors differ on the maximum error: Hastings says it is 1.5E-7, the Handbook of Mathematical Functions[6] says it is 7.5E-8, and the IBM Scientific Subroutine Package says it is 7E-7.

$$P(x) = 1 - Z(x)(b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5)$$

Where:

$t = 1/(1+px)$

$Z(x) = e^{-x^2/2}/\sqrt{2\pi}$

A line of code truncating the argument at =—6 avoids overflow on 32 bit single precision computers. Values of x outside this range yield probabilities sufficiently close to 1 or 0 to be ignored in most practical problems.

```
      REAL FUNCTION NDTR (x)
c     Integrate normal distribution from -infinity to x.
c     Source: Adapted from ndtr in the IBM SSP pg78.

      ax = abs(x)
      t = 1.0/(1.0+.2316419*ax)
      d = 0.0
      if(ax.lt.6.0)d = 0.3989423*exp(-x*x/2.0)
      p = 1.0 - d*t*((((1.330274*t - 1.821256)*t +
     *1.781478)*t - 0.3565638)*t + 0.3193815)
      if (x.lt.0.0) p = 1.-p
      ndtr = p
      END
```

5.2 Routine PRE - Print Error Budget

Routine PRE prints the errors stored in the intermediate data tables. This is the error budget data. At the beginning of each row of data, the routine prints the type of data in the row of the table.

```
      SUBROUTINE PRE
c     Print error budget table.
      common /ranges/ r(20), n
      common /budget/ h(20,20), v(20,20)
      character title(2,20)*24
      data title /
     1' 1 FIXED BIAS',' 1 FIXED BIAS',
     2' 2   Parallax',' 2   Parallax',
```

6. Handbook of Mathematical Functions, Abramowitz, M., and Stegun, Irene A., Dover Publications, Inc., NY, p 932, eq 26.2.17.

```
     3' 3   Drift',' 3   Drift',
     4' 4 VARIABLE BIAS',' 4 VARIABLE BIAS',
     5' 5   Cant',' 5   Muzzle Velocity error',
     6' 6   Crosswind',' 6   Rg est & PDC',
     7' 7   Jump',' 7   Jump',
     8' 8   Fire Control',' 8   Fire Control',
     9' 9   Boresight',' 9   Boresight',
     a'10   Plx, drf comp (PDC)','10   Range wind',
     b'11   Rotation of Earth','11   Air temp',
     c'12','12   Air density',
     d'13','13   Windage jump',
     e'14','14   Optical path bend',
     f'15','15   Vertical cant','16','16','17','17',
     h'18 RANDOM ERROR','18 RANDOM ERROR',
     i'19   Dispersion','19   Dispersion',
     j'20   Lay error','20   Lay error'/
1     format(a)
2     format(a12,12x,20f8.0)
3     format(a24,20f8.4)

      print *
      print 1, 'HORIZONTAL ERRORS'
        print 2, 'RANGE (m)', (r(i),i=1,n)
        DO 40 j=1,10
          print 3, title(1,j), (h(i,j),i=1,n)
40      CONTINUE
        DO 45 j=18,20
          print 3, title(1,j), (h(i,j),i=1,n)
45      CONTINUE
      print *
      print 1, 'VERTICAL ERRORS'
        print 2, 'RANGE (m)', (r(i),i=1,n)
        DO 50 j=1,15
          print 3, title(2,j), (v(i,j),i=1,n)
50      CONTINUE
        DO 55 j=18,20
          print 3, title(2,j), (v(i,j),i=1,n)
55      CONTINUE
      END
```

5.3 Routine PRTBL - Print Summary Results

PRTBL copies the relevant numbers from the error budget table to the output table, finds hit probabilities for the last column of the output table, and prints the table of final results in a form suitable for input to the tbl and troff processors.

Define the following variables.

Value	Comment
n	Number of ranges
r_i	Range to target (m)
t_{ij}	Table for storing results.
$t_{i4} = \sqrt{(t_{i2}^2 + t_{i3}^2)}$	Horizontal total dispersion (mils)
$t_{i8} = \sqrt{(t_{i5}^2 + t_{i7}^2)}$	Vertical total dispersion (mils).
$c_m = 3200/\pi$	Mils per radian.
$s = 1.15 c_m / r_i$	Angular distance from center to edge of target (mils).
$h_2 = (s - h_{i,1})/\sigma_{i,4}$	Normalized upper horiz limit of tgt.
$h_1 = (-s - h_{i,1})/\sigma_{i,4}$	Normalized lower horiz limit of tgt.
$v_2 = (s - v_{i,1})/\sigma_{i,8}$	Normalized upper vert limit of tgt.
$v_1 = (-s - v_{i,1})/\sigma_{i,8}$	Normalized lower vert limit of tgt.
$p_h = \frac{1}{\sqrt{2\pi}} \int_{h_1}^{h_2} e^{-y/2}\, dy$	Probability round is within horizontal limits of target.
$p_v = \frac{1}{\sqrt{2\pi}} \int_{v_1}^{v_2} e^{-z/2}\, dz$	Probability round is within vertical limits of target.
$t_{i9} = p_v p_h$	Probability of hitting the target.

```
      SUBROUTINE PRTBL(flag)
c     Print summary table of errors in mils.
      common /ranges/ r(20), n
      common /output/ table(20,9)
      common /budget/ h(20,20), v(20,20)
      logical flag(4)
      real ndtr
1     format(a)
2     format((f6.0,8('',f7.4),'',f7.3))
3     format(f8.0,8f8.4,f8.3)

      cm = 3200./3.1416
      DO 41 i=1,n
        table(i,1) = h(i,1)
        table(i,2) = v(i,1)
        table(i,3) = h(i,18)
        table(i,4) = h(i,4)
        table(i,5) = sqrt(h(i,4)**2 + h(i,18)**2)
```

```
        table(i,6) = v(i,18)
        table(i,7) = v(i,4)
        table(i,8) = sqrt(v(i,4)**2 + v(i,18)**2)
c       Find hit probability.
          s = 1.15*cm/r(i)
          h2 = (s-h(i,1))/table(i,5)
          h1 = (-s-h(i,1))/table(i,5)
          v2 = (s-v(i,1))/table(i,8)
          v1 = (-s-v(i,1))/table(i,8)
        ph = ndtr(h2) - ndtr(h1)
        pv = ndtr(v2) - ndtr(v1)
        table(i,9) = ph*pv
41    CONTINUE
      if (flag(3)) print 3, (r(i),(table(i,j),j=1,9),i=1,n)
      IF (flag(4)) THEN
      print*
      print 1, '.ce 3',
     1'Table ___. Ballistic Errors for a _______ Round Fired from a',
     2'Stationary _______ with a _______ Range finder and a',
     3'Computer','.TS','center box;','css|css|css|c','c|cs|ccc|ccc|c',
     4'c|cc|ccc|ccc|c','n|nn|nnn|nnn|n.',
     1'Horizontal (mils)Vertical (mils)','_',
     2'RangeFixed Bias (mils)RanVar.TotalRanVarTotal$P sub h#',
     3'(m)horiz.vert.errorbiasdisp.errorbiasdisp.',
     4'_'
      print 2, (r(i),(table(i,j),j=1,9),i=1,n)
      print 1, '.TE'
      ENDIF
      END
```

5.4 Routine RDTBL - Read Input and Set to Zero, Constant, or Interpolate.

If RDTBL is called with $m = 0$, it does nothing, so the jth lines of the error budget tables continue to contain zero values. If it is called with $m = 1$, it reads one horizontal angular error and one vertical and places these values in the jth rows of the horizontal and vertical error budget tables. If it is called with $m > 1$, it reads a table with 3 rows and m columns. The first row is the range, the second is horizontal angular errors, and the third is vertical angular errors. It then interpolates in this table for the desired ranges and places the interpolated values in the error budget tables.

The routine simply reads values and if m=1, copies the values to the appropriate rows. If m=2, it does a linear interpolation. The key variables and values computed are as follows:

Value	Comment
j	Index of row to be filled.
m	Copy if $m = 1$, interpolate if $m = 2$.
str	Character string describing error component.
echo	Logical controlling printing of input echo.
x_k	Target range values read in (m).
y_k	Horizontal errors read in (mils).
z_k	Vertical errors read in (mils).
r_i	Target ranges desired (m).
k	Index such that $x_k \leq r_i < x_{k+1}$.
$y = y_k + (x - x_k)(y_{k+1} - y_k)/(x_{k+1} - x_k)$	Std linear interpolation formula.
$h_{i,j} = y_k + (r_i - x_k)(y_{k+1} - y_k)/(x_{k+1} - x_k)$	Substituting for x, y.
$v_{i,j} = z_k + (r_i - x_k)(z_{k+1} - z_k)/(x_{k+1} - x_k)$	And similarly for z.

```
      SUBROUTINE RDTBL (j,m,str,echo)
c     Read constant values, or read & interpolate in table.
      common /budget/ h(20,20), v(20,20)
      common /ranges/ r(20), n
      character str*20
      logical echo
      real x(20), y(20), z(20)
      integer m, i, j, n
1     format(10f8.3)
2     format(10f8.0)
3     format(a)
4     format(2f8.3,'',a)

      if (echo .and. m.ne.1) print 3, str
      IF (m.lt.0) THEN
        print *, 'Error: Cannot use negative value.'
        STOP
      ELSEIF (m.eq.0) THEN
c       Leave jth row zero.
      ELSEIF (m.eq.1) THEN
c     Set row of errors to a constant.
        read *, y(1), z(1)
        if (echo) print 4, y(1), z(1), str
        DO 40 i=1,n
          h(i,j) = y(1)
          v(i,j) = z(1)
40      CONTINUE
      ELSE
c     Read and interpolate in table.
        if (m.gt.20) print 3, 'Warning: only 20 values used'
        if (m.gt.20) m=20
```

```
        read *, (x(i),i=1,m)
        read *, (y(i),i=1,m)
        read *, (z(i),i=1,m)
        IF (echo) THEN
          print 2, (x(i),i=1,m)
          print 1, (y(i),i=1,m)
          print 1, (z(i),i=1,m)
        ENDIF
        DO 50 i=1,n
          call hunt(x,m,r(i),k)
          h(i,j) = y(k)+(r(i)-x(k))*(y(k+1)-y(k))/(x(k+1)-x(k))
          v(i,j) = z(k)+(r(i)-x(k))*(z(k+1)-z(k))/(x(k+1)-x(k))
50      CONTINUE
      ENDIF
      END
```

INTENTIONALLY LEFT BLANK.

Appendix A. Comparison of TAM and PH1

The TAM program is based on PH1, an older program used by AMSAA. For all practical purposes, TAM produces the same numbers. It does not calculate zeroing errors associated with obsolete tanks. In all other respects, it is far superior to PH1. So says the author of TAM.

Accuracy. Both produce essentially the same numbers except that TAM does not produce certain zeroing errors associated with older tanks. Both programs were exercised with four test cases. The critical test is whether TAM produces the same hit probabilities PH1 produces. It does to three decimal places. Occasionally TAM generates error values which differ from PH1's by up to five thousands of a mil. This is an insignificant error of 15 millimeters at three kilometers. This difference appears to be due to differences in the time of flight and super-elevation calculated by the two programs. PH1 is slightly more accurate because it adjusts for slight differences in air density.

Ease of use. TAM is documented but PH1 documentation is only in the initial stages of writing. This is probably the most important factor in making TAM easier to use. TAM requires one page of input; PH1 requires three, some of which never changes. TAM reads input in free format; PH1 requires values in fixed fields so it is not forgiving of alignment errors. Target ranges input to TAM are in the form of a specific list of ranges; PH1 target ranges are specified by two cryptic lists of indexes. TAM produces all error budget data in two tables, one for horizontal errors and the second for vertical errors. PH1 produces the fixed bias numbers and random error output in a separate table in a different format.

Output. Although both programs produce the same output with or without lay error, TAM has an option to produce it in typesetter quality. Both produce the information found in TAM's intermediate error budget tables but PH1 does not produce it in the same compact form.

Programming Style. PH1 is an older program and does not incorporate modern programming principles; TAM was written using top-down structured principles. It is modular, with indentation to aid readability. Table A1 summarizes some of the advantages TAM has over PH1.

TABLE 10. Comparison of PH1 and TAM

	PH1	TAM	REDUCTION
Lines of code	2,600	650	75%
Variables	900	90	90%
Mean length of routines (lines)	650	41	94%
Max length of routines (lines)	2318	84	96%
GOTO's	420	4	99%
Hardware	Super computer	PC to super computer	

Compiler	Fortran 77	Fortran 77
Indentation	No	Yes
Documentation	No	Yes

Appendix B. Preparing Drag Data

Drag data appears in various formats, some not suitable for input to TAM. The two brief programs presented below reformat the data in a form acceptable to TAM.

Very often, drag data appears in the form shown below:

```
 58000000+00 00000000+00                                50000000+0083001045
 75500000+00-35000000+00                                60000000+0083002045
 60500000+00-10000000+00                                70000000+0083003045
 46500000+00 10000000+00                                80000000+0083004045
 26500000+00 35000000+00                                90000000+0083005045
-14900000+01 23000000+01                                10000000+0183006045
 10000000-01 80000000+00                                10500000+0183007045
 74500000+00 10000000+00                                11000000+0183008045
 96500000+00-10000000+00                                11500000+0183009045
 13100000+01-40000000+00                                12000000+0183010045
 11500000+01-26666667+00                                15000000+0183011045
 10350000+01-19000000+00                                20000000+0183012045
 99500000+00-17000000+00                                25000000+0183013045
 87000000+00-12000000+00                                30000000+0183014045
 84000000+00-11000000+00                                40000000+0183015045
```

Each line contains the following information: Columns 1-72 contain six fields of 12 columns each. Columns 73-75 contain a three digit code identifying the round. Columns 76-77 contain the line count. Columns 78-79 appear to be unused. Column 80 contains a code describing the data. If the code is '5' the data is K_D data and if it is '6', the data is C_D.

Our Gould Fortran 77 compiler (and perhaps others) is not able to read the numbers in fields 1-6. The following program inserts decimal points and an 'e' before the exponent so that it is readable.

```
c     Insert decimal point & e in exponent of drag numbers.
      character a(6)*12, b*8, spac12*12
1     format(6a12,a8)

      spac12='            '
20    CONTINUE
        read(5,1,end=99) a, b
        DO 30 i=1,6
          IF (a(i).ne.spac12) THEN
            a(i)(9:9) = 'e'
            a(i)(8:8) =  a(i)(7:7)
            a(i)(7:7) =  a(i)(6:6)
            a(i)(6:6) =  a(i)(5:5)
            a(i)(5:5) =  a(i)(4:4)
            a(i)(4:4) =  a(i)(3:3)
```

```
            a(i)(3:3) =  a(i)(2:2)
            a(i)(2:2) = '.'
          ENDIF
30      CONTINUE
        print 1, a, b
      GOTO 20
99    CONTINUE
      END
```

The output looks like this:

```
 .580000e+00 .000000e+00                                    .500000e+0083001045
 .755000e+00-.350000e+00                                    .600000e+0083002045
 .605000e+00-.100000e+00                                    .700000e+0083003045
 .465000e+00 .100000e+00                                    .800000e+0083004045
 .265000e+00 .350000e+00                                    .900000e+0083005045
-.149000e+01 .230000e+01                                    .100000e+0183006045
 .100000e-01 .800000e+00                                    .105000e+0183007045
 .745000e+00 .100000e+00                                    .110000e+0183008045
 .965000e+00-.100000e+00                                    .115000e+0183009045
 .131000e+01-.400000e+00                                    .120000e+0183010045
 .115000e+01-.266666e+00                                    .150000e+0183011045
 .103500e+01-.190000e+00                                    .200000e+0183012045
 .995000e+00-.170000e+00                                    .250000e+0183013045
 .870000e+00-.120000e+00                                    .300000e+0183014045
 .840000e+00-.110000e+00                                    .400000e+0183015045
```

Denote the columns c_1 ... c_5, and the sixth as m. Columns one to five are coefficients and the sixth is the mach number. The drag coefficient C_d is:

$$C_d = \sum_{1}^{5} c_i m^{i-1}$$

This is still not in a form acceptable to TAM. So the data is read by the following program.

```
c     Convert drag data to format for TAM and ARL's PH1.
      real m
1     format(6g12.4)
2     format(2f8.3)

20    CONTINUE
        read (5,1,end=99) a,b,c,d,e,m
        cd = a+m*(b+m*(c+m*(d+m*e)))
        print 2, m, cd
      GOTO 20
99    CONTINUE
      END
```

Now, the data is in a form TAM can use. It looks like this:

```
0.500   0.580
0.600   0.545
0.700   0.535
0.800   0.545
0.900   0.580
1.000   0.810
1.050   0.850
1.100   0.855
1.150   0.850
1.200   0.830
1.500   0.750
2.000   0.655
2.500   0.570
3.000   0.510
4.000   0.400
```

INTENTIONALLY LEFT BLANK.

No. of Copies	Organization
2	Administrator Defense Technical Info Center ATTN: DTIC-DDA Cameron Station Alexandria, VA 22304-6145
1	Commander U.S. Army Materiel Command ATTN: AMCAM 5001 Eisenhower Ave. Alexandria, VA 22333-0001
1	Director U.S. Army Research Laboratory ATTN: AMSRL-D 2800 Powder Mill Rd. Adelphi, MD 20783-1145
1	Director U.S. Army Research Laboratory ATTN: AMSRL-OP-CI-AD, Tech Publishing 2800 Powder Mill Rd. Adelphi, MD 20783-1145
2	Commander U.S. Army Armament Research, Development, and Engineering Center ATTN: SMCAR-IMI-I Picatinny Arsenal, NJ 07806-5000
2	Commander U.S. Army Armament Research, Development, and Engineering Center ATTN: SMCAR-TDC Picatinny Arsenal, NJ 07806-5000
1	Director Benet Weapons Laboratory U.S. Army Armament Research, Development, and Engineering Center ATTN: SMCAR-CCB-TL Watervliet, NY 12189-4050
(Unclass. only) 1	Commander U.S. Army Rock Island Arsenal ATTN: SMCRI-IMC-RT/Technical Library Rock Island, IL 61299-5000
1	Director U.S. Army Aviation Research and Technology Activity ATTN: SAVRT-R (Library) M/S 219-3 Ames Research Center Moffett Field, CA 94035-1000

No. of Copies	Organization
1	Commander U.S. Army Missile Command ATTN: AMSMI-RD-CS-R (DOC) Redstone Arsenal, AL 35898-5010
1	Commander U.S. Army Tank-Automotive Command ATTN: ASQNC-TAC-DIT (Technical Information Center) Warren, MI 48397-5000
1	Director U.S. Army TRADOC Analysis Command ATTN: ATRC-WSR White Sands Missile Range, NM 88002-5502
1	Commandant U.S. Army Field Artillery School ATTN: ATSF-CSI Ft. Sill, OK 73503-5000
(Class. only) 1	Commandant U.S. Army Infantry School ATTN: ATSH-CD (Security Mgr.) Fort Benning, GA 31905-5660
(Unclass. only) 1	Commandant U.S. Army Infantry School ATTN: ATSH-CD-CSO-OR Fort Benning, GA 31905-5660
1	WL/MNOI Eglin AFB, FL 32542-5000
	Aberdeen Proving Ground
2	Dir, USAMSAA ATTN: AMXSY-D AMXSY-MP, H. Cohen
1	Cdr, USATECOM ATTN: AMSTE-TC
1	Dir, ERDEC ATTN: SCBRD-RT
1	Cdr, CBDA ATTN: AMSCB-CI
1	Dir, USARL ATTN: AMSRL-SL-I
10	Dir, USARL ATTN: AMSRL-OP-CI-B (Tech Lib)

No. of
Copies Organization

Aberdeen Proving Ground

7 Dir, USAMSAA
ATTN: AMXSY-G,
Wilbert Brooks
Ed Christman
Barry Siegal
Richard Norman
Ed Walker
Mary Ritondo
AMXSY-A, John Meredith

USER EVALUATION SHEET/CHANGE OF ADDRESS

This Laboratory undertakes a continuing effort to improve the quality of the reports it publishes. Your comments/answers to the items/questions below will aid us in our efforts.

1. ARL Report Number ARL-MR-48 Date of Report March 1993

2. Date Report Received _____

3. Does this report satisfy a need? (Comment on purpose, related project, or other area of interest for which the report will be used.) _____

4. Specifically, how is the report being used? (Information source, design data, procedure, source of ideas, etc.) _____

5. Has the information in this report led to any quantitative savings as far as man-hours or dollars saved, operating costs avoided, or efficiencies achieved, etc? If so, please elaborate. _____

6. General Comments. What do you think should be changed to improve future reports? (Indicate changes to organization, technical content, format, etc.) _____

CURRENT ADDRESS

_____
Organization

_____
Name

_____
Street or P.O. Box No.

_____
City, State, Zip Code

7. If indicating a Change of Address or Address Correction, please provide the Current or Correct address above and the Old or Incorrect address below.

OLD ADDRESS

_____
Organization

_____
Name

_____
Street or P.O. Box No.

_____
City, State, Zip Code

(Remove this sheet, fold as indicated, staple or tape closed, and mail.)

DEPARTMENT OF THE ARMY

OFFICIAL BUSINESS

BUSINESS REPLY MAIL
FIRST CLASS PERMIT No 0001, APG, MD

Postage will be paid by addressee.

Director
U.S. Army Research Laboratory
ATTN: AMSRL-OP-CI-B (Tech Lib)
Aberdeen Proving Ground, MD 21005-5066

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES